

AIAA 2002-0380

Toward An Efficient Icing CFD Process
Using An Interactive Software Toolkit –
Smagglce 2D

Mary B. Vickerman, Yung K. Choo, Herbert W. Schilling,
Marivell Baez, Donald C. Braun, Barbara J. Cotton

NASA Glenn Research Center
Cleveland, Ohio

40th AIAA Aerospace Sciences Meeting & Ex-
hibit

14-17 January 2002 / Reno, NV

TOWARD AN EFFICIENT ICING CFD PROCESS USING AN INTERACTIVE SOFTWARE TOOLKIT – *SmagIce 2D*

Mary B. Vickerman, Yung K. Choo, Herbert W. Schilling, Marivell Baez, Donald C. Braun, Barbara J. Cotton

NASA Glenn Research Center
Cleveland, Ohio 44135

Abstract

Two-dimensional CFD analysis for iced airfoils can be a labor-intensive task. The software toolkit *SmagIce 2D* is being developed to help streamline the CFD process and provide the unique features needed for icing. When complete, it will include a combination of partially automated and fully interactive tools for all aspects of the tasks leading up to the flow analysis: geometry preparation, domain decomposition, block boundary discretization, gridding, and linking with a flow solver. It also includes tools to perform ice shape characterization, an important aid in determining the relationship between ice characteristics and their effects on aerodynamic performance. Completed tools, work-in-progress, and planned features of the software toolkit are presented here.

Introduction

Analysis of the aerodynamic performance of wings with ice accretion is important because ice degrades lift and increases drag, increasing the possibility of a safety threat. Although Computational Fluid Dynamics (CFD) tools are available for three-dimensional (3D) aerodynamic analysis, their application to iced wings is very difficult. The analysis of iced wings requires surface definitions that can be derived from point cloud data obtained by laser scanning, but due to the irregular geometric and optical properties of ice, the scanning often introduces significant noise into the scanned data. Furthermore, three-dimensional surface and volume grid generation for CFD analysis using existing tools is extremely difficult because the irregular ice surface contains various conic shapes, varying degrees of surface roughness, and cavities. In addition, the 3D flow simulation for iced wings is computationally very expensive today. Because of these reasons, a very limited

3D analysis was performed to supplement a thorough 2D analysis¹. Until the 3D analysis becomes easier and faster for icing effects study on wing performance, two-dimensional analysis will play an important role in complementing experimental studies and providing insights to effects of ice on airfoil aerodynamics^{1,2,3,4}.

Two-dimensional CFD analysis can also be labor-intensive and computation-intensive, depending on ice shape characteristics and the tools used. The current choices for grid generation on iced airfoils lie at two ends of the spectrum: fully automatic or fully interactive. Fully automatic methods (such as Thompson/Soni^{5,6}) are needed when used with other tools such as the ice accretion code LEWICE⁷ and are effective for relatively moderate ice shapes. But for applications where highly irregular ice surfaces are studied, or when user-control of the grid is important, interactive tools are desirable. However, general-purpose, interactive grid generation tools (such as GRIDGEN⁸) are not customized for ice shapes, and so require extensive experience and effort⁹ on the part of the researcher to generate appropriate grids. Furthermore, these general-purpose tools may require additional work: a pre-processing step of geometry preparation to remove input errors and apply smoothing to the ice, and, if multi-block grids are used, a post-processing step to define connectivity of the grid blocks.

Software is needed to make the entire icing CFD process easier and more efficient than the current manual process, while still providing reliable results. *SmagIce 2D* is a software tool kit that is being developed at NASA Glenn for this purpose. The goal is to streamline the CFD process and provide the unique features needed for icing. In Version 1.0, support was provided for “ice shape characterization and control,” using interactive tools¹⁰. For icing CFD, ice shape characterization, examining the data, and preparation of iced airfoil geometry for grid generation are important in the quest for determining the relationship between ice characteristics and their effects on aerodynamic performance. Version 1.2, which is close to release, brings the capabilities of Version 1.0 (for UNIX plat-

Copyright © 2001 by the American Institute of Aeronautics and Astronautics, Inc. No copyright is asserted in the United States under Title 17, U.S. Code. The U.S. Government has a royalty-free license to exercise all rights under the copyright claimed herein for Government Purposes. All other rights are reserved by the copyright owner.

forms) to the MS-Windows platform, with additional capabilities for ice shape characterization, geometry modification, and adding artificial ice shapes to an airfoil. In Version 2.0, which is under development, the entire “icing aerodynamic analysis,” process, as shown in Figure 1, will be streamlined. To accomplish this, SmagglIce will partially automate many common tasks in the manual process, as well as provide a set of interactive tools for the user. The tools will cover the range of tasks required for icing CFD: geometry preparation, domain decomposition, block modification, grid generation and modification, and linking with the CFD flow solver. Details of this work in progress are presented here. Version 2.0 is intended to provide a balance between various needs for reliability, versatility, efficiency, and usability. Therefore, each technology component is first weighed by its contribution to the overall CFD process before being incorporated into the code.

SmagglIce 2D is designed for 2D airfoils with ice accretion. It supports multi-block, structured grids. The grid blocks in general are abutting, although an overlap area may be specified between the near-field blocks and the outer block. Abutting blocks do not have to share corner points and may have either 1-to-1 matching or mismatched grids at the block boundaries. These block interfaces are those supported in the CFD General Notation System (CGNS)¹¹.

Version 1.2 Capabilities

SmagglIce 2D provides Graphical User Interface (GUI) controls for access to its automatic and interactive tools. The user interface consists of a primary Main Window and additional sub-windows. The SmagglIce Main Window (Figure 2) includes the following distinctive areas: Menu Bar, Graphics Drawing Area, Tool Bar, Information, Current Object Info, and Graphics Window Mode. These are similar to the interface descriptions of SmagglIce version 1.0¹⁰. All functions are accessed through GUI controls, which are sensitized at appropriate times to guide the user through program operation. The GUI also checks and validates user input, prompting for corrections as needed, and provides on-line context-sensitive help. Help is provided through HTML files and can be accessed through the Menu Bar, from the Help pushbutton on individual sub-windows, or directly from a web browser.

Tools in version 1.2 include two types. Ice shape characterization tools are used to measure and record location, length, angle, arc length, and ice area. This gives users the means to measure the physical characteristics of ice such as icing limit locations, horn height and angle, distance along the clean airfoil from the leading edge to a prominent ice location, or the area of ice between two user-specified points (Figure 3). Geometry

preparation tools are used to detect and correct deficiencies or errors (e.g., twists) in the input ice shape, perform controlled surface smoothing to a level that will make the CFD process manageable, rotate/translate airfoil components, and add artificial ice shapes to an airfoil.

The artificial ice creation tool provides users with geometric shapes that can be attached to the surface of the clean airfoil, facilitating studies of the effects various ice shapes have on the aerodynamic performance. The *Add Artificial Ice* window (Figure 4) allows the user to select one of the geometric shapes to add: right triangle (forward-facing or backward-facing), generic triangle, rectangle, quarter circle (forward-facing or backward-facing), semi-circle, or trapezoid. Parameters for location, replication, size and number of points may be set. For example, the *X Location* gives the value (in chord units) where the geometric shape will be placed, and it can be put on either the upper or lower surface of the airfoil. A train of shapes can be added easily by specifying a number of shapes and the space between each one. The size of the shape is specified by parameters such as width, length, and angle, where the angle is measured in degrees relative to the normal to the airfoil at the shape location. A positive angle will tilt the shape upstream, while a negative angle will tilt downstream. Finally, the numbers of points along the edges of the shapes are specified, with different shapes requiring different parameters. For example, for a triangle, the user will need to specify the number of points on the upstream edge and the downstream edge; for a quarter-circle, the number of points on the arc is needed.

Version 2.0 Overview

Work in progress for version 2.0 includes tools for domain decomposition, block discretization, block modification, grid generation and evaluation, and linking with a CFD flow solver.

To perform the domain decomposition, first the wake is defined, then a viscous sublayer block (proposed by Shim, Chung, and Lee¹²) is created based on user-specified parameters. The viscous sublayer is a C-shaped block that wraps around the iced airfoil and extends back along the wake to the exit boundary. It provides a simplified boundary condition specification, as well as slight smoothing of the block edges shared with adjacent blocks. Next, automatic initial blocking is done based on the actual geometry of the iced airfoil and the user-selected ice shape class¹². The ice shape class is a generalized definition of the initial block topology to be used for the near field: the number of blocks, their relative locations, and their connectivity to each other. Critical points on the iced airfoil determine how the block topology is applied to the particular ice shape being studied (i.e., where corner points of blocks

will be located). These critical points are automatically detected but may be modified by the user. For block boundary discretization, points are automatically distributed along the block edges based on the point distribution and shape of the ice. Finally, an outer block, which covers the far field of the domain, is created based on user-specified parameters. Figure 5 shows an iced airfoil and its full domain, including the near field (which is decomposed further based on the ice shape class) and the overlapping outer block.

User-initiated block modification will be made available to meet the needs of flow characteristics around ice. Support will be provided for interactive dividing and merging of blocks to accommodate feathers and cavities. Users will also be able to modify edge meshing and automatically propagate those point distributions to other block edges. Those may be shared edges of adjacent blocks, the opposite edge of the same block, or recursively extending outward from the airfoil through all blocks.

Grid tools will include automatic definition of boundary conditions, grid generation, automatic computation of connectivity between adjacent or overlapping blocks, interactive grid refinement for grid sensitivity study, and visual feedback indicating grid quality. Exact grid quality measures include: grid spacing, ratio of consecutive grid spacings, grid aspect ratios, grid area, and grid line angle. Approximate measures include: grid Jacobian, grid smoothness integral, grid orthogonality integral, and grid volume adaption integral.

CGNS^{13,14} files will be used to store and communicate grids, boundary conditions, connectivity, and solutions. This will allow output from SmagglIce 2D to be used directly as input into flow solvers that support CGNS. In particular, SmagglIce 2D will be closely tied with the CFD flow solver WIND^{15,16}.

Domain Decomposition and Blocking

Blocks are generated during domain decomposition. Each block has four edges and four corners. A block can contain a grid and optionally a solution. Blocking is accomplished by taking as input the airfoil/ice points, the ice shape class, critical points, and domain parameters, and generating the block boundary points as output. Work in the areas of domain decomposition and blocking is currently focused on designing the user interface and determining effective algorithms for automatically performing the initial blocking of an iced airfoil.

User Interface

The process of domain decomposition and initial block discretization, which is a subset of version 2.0, is presented here from a user's perspective. Within SmagglIce

2D, the user reads the element data. Elements represent boundaries of solid objects, usually airfoils or ice shapes. Element data can be read in from three types of files: Icing Research Tunnel (IRT) data files, LEWICE output files, or SmagglIce formatted files. Often, two elements will be read: an ice shape and its associated clean airfoil. The user must specify which element will be used as the reference airfoil, because that is used to determine the chord length, to which blocking parameters are normalized. If the ice element is only partial, it is extended by copying the points of the reference airfoil element into the ice shape element to form a complete iced airfoil element. The iced airfoil element can be prepared for blocking through curve modification and smoothing: discretization, tanh redistribution, and free-form manipulation. Once the iced airfoil is ready for blocking the user will perform domain decomposition, which includes four ordered steps: wake definition, viscous sublayer definition, near field decomposition, and outerblock definition. All steps can be accessed from the "Block,, menu in the main window, with each one bringing up its own sub-window containing pre-defined parameters that the user may modify. The blocks generated with the parameters are displayed in the graphics window and can either be accepted or modified further by the user. Pressing "Restore Defaults,, will reset all of the parameters to the system default values.

The *Wake Definition* window allows the user to specify parameters for the wake: number of points (default=40), length in chord units (default=15) and angle (default=0). Once the wake has been defined, the Graphics Window will display the wake.

The *Viscous Sublayer* window allows the user to specify parameters for the viscous sublayer block: number of J points (default=15), thickness (default=.001) and minimum grid spacing (default=.00001). If concavities in the ice shape, in combination with the sublayer thickness, introduce tangles into the block edges, these can be removed by: reducing the thickness of the viscous sublayer, manually smoothing the ice surface, or using an automatic process to slightly modify the ice surface and smooth the tangles. All tangles must be removed in order to continue with the next step of Near Field Decomposition.

The *Near Field Decomposition* window (Figure 6) has several functions. The user will choose from one of the pre-defined ice shape class topologies. Currently there are five, with more to follow as needed: "Clean Airfoil / Single Block,, "Single Horn / 2 Blocks,, "Single or Double Horn / 3 Blocks,, "Double Horn with Cavity / 4 Blocks,, and "Three Horn / 5 Blocks,, (Figure 7). After a class is selected, a representative image of that class will be displayed, the number of blocks are defined, the block boundaries are generated and

discretized, the critical points are automatically selected and the Graphics Window displays the near field decomposition. The surrounding block parameters for the near field are: number of J points and radius. The “Move Critical Points ...”, pushbutton will allow the user to change the location of the critical points.

The *Outerblock Definition* window allows the user to specify parameters for the outer block: number of overlap cells in outer block (default=2) and near field (default=2), number of points in I direction (default=370) and J direction (default=20), and length of the outer radius (default=15).

Algorithms and Approaches

The viscous sublayer block is created based solely on the iced airfoil points and wake definition. The initial outer boundary of the viscous sublayer block is generated by creating points the specified distance away and normal to the iced airfoil. If tangles are to be automatically removed, the process is to: find the point of intersection in the outer boundary, use the three boundary points on either side of the point of intersection as control points, generate a smooth curve using those control points, project a normal of the specified thickness back towards the airfoil to modify the inner layer, and smooth at the endpoints of the inner layer segment (Figure 8). Finally, creases in the boundaries are removed by smoothing using five points centered at the crease. This process will remove small, narrow cavities in the ice shape. Alternatively, the user may choose to model these cavities rather than removing them, which will be accomplished by further splitting of the viscous sublayer block.

The concept of ice shape classes is used to make it easy for users to select an initial blocking scheme that requires very little user input. The user chooses a class that is representative of the actual ice shape being blocked. The class chosen determines the number of initial blocks, where their corners are located, and their connectivity to each other. This approach separates the blocking topology from the ice geometry, so that a single ice shape class can be used for many different iced airfoils, as long as their shapes are similar. After initial blocking, users can modify the blocks by splitting and merging, allowing for any number and connectivity of blocks. Five classes have been defined so far. For example, Figure 9 shows the near-field decomposition for class “Single or Double Horn / 3 Blocks,,,” which includes, besides the viscous sublayer block and outer block, 3 other blocks which make up the near field: one below the airfoil behind the lower horn, one above the airfoil behind the upper horn, and one surrounding those two blocks as well as the area at the leading edge. Figure 10 shows the near-field decomposition for class “Three Horn / 5 Blocks,,,”

Critical points determine the locations of corners of blocks needed for domain decomposition of the near field. These critical points are chosen automatically, but the user has the option of moving or reselecting them if needed. The number and relative locations of the critical points for the domain decomposition will be based on the chosen ice shape class. The initial locations of those critical points can be estimated by extracting information from the iced airfoil’s convex hull (Figures 11 and 12):

- 1 - Using the left-most third of the iced airfoil (leading edge), generate the convex hull.
- 2 - For each point on this section of airfoil, find distance to the convex hull.
- 3 - Find the maximum distance. That is the most prominent concave critical point.
- 4 - Find the points to either side that have a distance of zero. Those are the convex critical points.
- 5 - Remove that segment (between the two zero values) from consideration, and repeat from step three for the number of blocks needed around the iced airfoil.
- 6 - This process will have defined three corner points of each of the blocks around the airfoil. If a fourth point is desired, another local peak within that segment is used. Finally, the rightmost and leftmost critical points (as calculated) are not actually used, since a point on the exit boundary is used instead.

To make it easy for users and SmagIce developers to define classes, we have defined a format for a domain decomposition file and some algorithms that use the file as input to generate block boundaries. A different domain decomposition file is used for each ice class. The block boundaries are defined by their boundary grid points. The goals of the domain decomposition file format are: easy to create, read, and understand; easy to parse and incorporate into SmagIce code; flexible enough to handle all classes and block types; requires minimal additional user input.

The domain decomposition is based on several concepts^{17,18}. Blocks are defined by their boundaries; each has four sides: bottom, top, left and right. The bottom side of a block is the side of the block closest to the airfoil/ice surface. Sides of a block are defined by points, not curves. The blocks and their boundaries are “built,, from the “foundation,, of the iced airfoil surface outward toward the outer boundary of the near field. A variety of different types of blocks are used to build a class. Each type of block is defined by: what sides are given as known sides, and how the remaining sides are determined. The blocks are created in the order in which they are defined in the domain decomposition file. This is important because the definitions of later blocks often make use of block sides defined and created earlier. The known side points can be taken from either: the airfoil/ice surface, the wake points, or side points of previously generated block boundaries. Some

of the types of blocks that have been implemented so far are listed below. Brief explanations are given of the algorithms used to determine the unknown sides of the blocks.

- *Viscous Sublayer*- Details given above.
- *LEFT Side on Exit Boundary*- Bottom and right side points are given. To compute top side: use convex hull of bottom points, offset outward, clean up. To compute left side: use distribution of points on right to distribute points on exit boundary between top and bottom sides.
- *RIGHT Side on Exit Boundary*- Same as LEFT Side on Exit Boundary, except reverse left and right.
- *LEFT and RIGHT Sides on Exit Boundary*- Bottom points are given. To compute top side: use outer boundary of near-field domain. Distribute points based on distribution of bottom points. To compute left and right sides: use stretching functions to distribute points on exit boundary between top and bottom sides.
- *Cavity*- Left, right and bottom points are given. To compute top side: create a line segment between the tops of the left and right sides and distribute points along the line based on the distribution of the points on the bottom side. To compute left and right sides: if the sides have equal numbers of points, we are done. If not, fit a spline through the side with the fewer points. Distribute points on this spline based on the number and distribution of points on the side with more points.
- *Surrounding Block - Radial Cut - LEFT Side on Exit Boundary*- Bottom points are given. To compute top side: use outer boundary of near-field domain. Distribute points based on distribution of bottom points. To compute left side: use stretching functions to distribute points on exit boundary between top and bottom sides. To compute right side: find the point on the outer boundary of the near field nearest to the rightmost point on the bottom. Create a line between these two points. Use a stretching function to distribute points along the line.
- *Surrounding Block - Radial Cut - RIGHT Side on Exit Boundary*- Same as Surrounding Block Radial Cut LEFT Side on Exit Boundary, except reverse left and right.
- *Surrounding Block Bottom, Left, Right Known*- Bottom, left and right points are given. To compute top side: use outer boundary of near-field domain. Distribute points based on distribution of bottom points.
- *Outer Block*- Bottom, left and right points on which to base bottom, left and right points of outer block are given. To compute the bottom side: the outer block is a C grid whose bottom side is slightly inside the outer boundary of the near field. The number of points along the bottom side is given by a user input parameter. Their distribution is similar to the distribution of the grid lines on the outer boundary of the near field. To compute the left and right sides:

the number of points is given by a user input parameter. They are distributed based on a stretching function. The minimum spacing is determined so that it is close to the spacing of the grid lines at the outer boundary of the near field. To compute the top side: use the outer boundary of outerblock domain. Distribute points based on distribution of bottom points.

Domain Decomposition File

Below is an example domain decomposition file for the ice shape class shown in Figure 9, "Single or Double Horn / 3 Blocks,.". The format of the file is a CLASS name followed by a series of BLOCK definition sections. Each BLOCK section has:

- an arbitrary name (e.g., in the first block below, the block name is "The Sublayer,,")
- the block type (e.g., in the first block below, the block type is "Viscous Sublayer,,")
- a series of assignment statements that define where the known side points come from (WAKEPOINTSLOWER, WAKEPOINTSUPPER, AIRFOIL, BLOCK, or a combination of all of them)
- the BLOCK term used on the right hand sides of the assignment statements define:
 - what existing block to take the points from
 - what side of the block to take the points from
 - between what critical points to take the points from. This is an optional parameter. (You can use FIRST or LAST to indicate the first and last points on a side.) Note that even though critical points are initially defined only on the airfoil surface, critical points are "propagated,, to other block sides if the point on that side is opposite a critical point or the point is a neighbor of a critical point on an abutting block.
- PARAMETERS sections for constant parameters needed for the algorithms that compute the unknown block side points

Example

```
CLASS "Single or Double Horn / 3 Blocks"

BLOCK "The Sublayer" TYPE "Viscous Sublayer"
  BOTTOM = WAKEPOINTSLOWER
  AIRFOIL ( FIRST : LAST )
  WAKEPOINTSUPPER
ENDBLOCK

BLOCK "Behind Lower Horn" TYPE "LEFT Side on Exit Boundary"
  BOTTOM = BLOCK("The Sublayer",TOP,1:LAST)
  RIGHT = BLOCK("The Sublayer",TOP,2:1)
ENDBLOCK

BLOCK "Behind Upper Horn" TYPE "RIGHT Side on Exit Boundary"
  BOTTOM = BLOCK("The Sublayer",TOP,FIRST:4)
  LEFT = BLOCK("The Sublayer",TOP,4:3)
ENDBLOCK
```

```

BLOCK "Surrounding" TYPE "LEFT and RIGHT
Sides on Exit Boundary"
  BOTTOM = BLOCK("Behind Lower Horn", TOP)
          BLOCK("The Sublayer", TOP, 3:2)
          BLOCK("Behind Upper Horn", TOP)
  BOTTOMSPACING = BLOCK("The Sublayer")
  PARAMETERS
    stretchingType = 2
  ENDPARAMETERS
ENDBLOCK

BLOCK "Outer block" TYPE "Outer block"
  BOTTOM=BLOCK("Surrounding", TOP,
FIRST:LAST)
  RIGHT=BLOCK("Surrounding",RIGHT,
FIRST:LAST)
  LEFT=BLOCK("Surrounding", LEFT, LAST:FIRST)
ENDBLOCK

```

Gridding Support

Grid generation methods being implemented include transfinite interpolation, Poisson, and elliptic. But current work related to gridding is mostly focused on two areas: automatic definition of boundary conditions and automatic computation of connectivity between adjacent or overlapping blocks.

In order to support simulation of subsonic turbulent flow over clean and iced airfoils while minimizing tedious user interaction, SmagglIce version 2.0 will automatically assign to each point of each edge of each block either a boundary condition or else connectivity to an adjacent or overlapping block. Downstream points along the outermost boundary of the flow domain will automatically be assigned a “subsonic outflow,, boundary condition. Other points along the outermost boundary of the flow domain will automatically be assigned a “freestream,, boundary condition so CFD analysis of local flow conditions can determine if the flow is into or out of the flow domain. Based on a single choice by the user, all block edge points along the ice / airfoil surface will automatically be assigned one of the following three boundary conditions: “inviscid flow,, where air can slip tangent to the surface and there is no heat transfer across the adiabatic surface; “viscous flow with heat flux,, where air cannot slip at the surface and heat transfer is either adiabatic or specified; or “viscous isothermal flow,, where air cannot slip at the surface and the temperature at the surface is a constant user-specified value.

SmagglIce version 2.0 will automatically assign appropriate connectivity information to each block edge point that is not along the ice / airfoil surface or along the outermost boundary. If subedges of two blocks lie on a common curve, the points along those subedges will be assigned either “one-to-one abutting,, or else “abutting mismatched,, connectivity, depending on whether or not the points of those two block subedges share exactly the same locations. Otherwise, if the gridded outer block overlaps points of a gridded inner block, Smag-

glIce will use inverse bilinear interpolation to compute “overlapping,, connectivity information for those overlapped points. In this latter case, SmagglIce will identify all anomalous special cases for inverse bilinear interpolation such as grid cells which are concave, are twisted (with opposite edges that intersect), or have three or four collinear vertices.

Link with CFD Flow Solver

CGNS is an emerging standard for storing, retrieving, and transferring data for CFD analysis in a format that can be written and read by various CFD programs running on a variety of computer architectures. CGNS has been developed and used by industrial, educational, and government organizations. SmagglIce version 2.0 will use CGNS files to store and communicate grids, boundary conditions, connectivity, and solutions in a format compatible with the WIND version 4 flow solver. In addition, SmagglIce will use an extension to CGNS to save and retrieve intermediate work, such as the distributions of points along block edges before grids are generated for those blocks.

Additional tools will facilitate linking with WIND. These will include tools to create the WIND input file, view convergence histories from runs, and display solutions from WIND that were stored in CGNS files.

References

1. Chung, J.J., Choo, Y., Reehorst, A., Potapczuk, M., Slater, J., “Navier-Stokes Analysis of the Flowfield Characteristics of an Ice Contaminated Aircraft Wing,, AIAA Paper 99-0375, Reno, NV, January 1999.
2. Chung, J.J., Addy, H., “A Numerical Evaluation of Icing Effects on a Natural Laminar Flow Airfoil,, AIAA Paper 2000-0096, Reno, NV, January 2000.
3. Addy, H., Chung, J.J., “A Wind Tunnel Study of Icing Effects on a Natural Laminar Flow Airfoil,, AIAA Paper 2000-0095, Reno, NV, January 2000.
4. Shim, J., Chung, J., Lee, K.D., “A Computational Investigation of Ice Geometry Effects on Airfoil Performance,, AIAA Paper 2001-0540, Reno, NV, January 2001.
5. Thompson, D.S., Soni, B.K., “ICEG2D (v2.0) - An Integrated Software Package for Automated Prediction of Flow Fields for Single-Element Airfoils with Ice Accretion,, NASA CR-2001-210965, 2001.
6. Choo, Y., Vickerman, M., Lee, K., Thompson, D., “Geometry Modeling and Grid Generation for Icing Effects and Ice Accretion Simulations on Airfoils,, Numerical Grid Generation in Computational Fluid Simulations on Airfoils, edited by B. Soni, J. Hauser, J. Thompson, and P.

- Eiseman, International Society for Grid Generation, Mississippi State, MS, 2000, pp. 1061-1070.
7. Wright, W. B., "A Summary of Validation Results for LEWICE 2.0,,," AIAA Paper 99-0249, AIAA 37th Aerospace Science Meeting, Reno, NV, January 1999.
 8. "GRIDGEN,,," February 16, 2001, <http://www.pointwise.com/prod/gg.htm/>, (May 24, 2001).
 9. Personal communication with James Joongkee (J.J.) Chung (April 2001): Experience based on the cases reported in AIAA Paper 2000-0097 by W. Wright and J.J. Chung.
 10. Vickerman, M., Choo, Y., Braun, D., Baez, M., Gnepp, S., "SmaggIce: Surface Modeling and Grid Generation for Iced Airfoils—Phase 1 Results,,," AIAA Paper 2000-0235, Reno, NV, January 2000.
 11. Allmaras, S., McCarthy, D., "CGNS Standard Interface Data Structures - Conventions,,," Oct. 25, 2001, <http://www.grc.nasa.gov/www/cgns/sids/conv.html#interfaces>, (Oct. 30, 2001).
 12. Shim, J., Chung J., Lee, K. D., "A Grid Generation Strategy for CFD Analysis of Iced Airfoil Flows,,," Numerical Grid Generation in Computational Fluid Simulations, edited by B. Soni, J. Hauser, J. Thompson, and P. Eiseman, International Society for Grid Generation, Mississippi State, MS, 2000, pp. 71-80.
 13. Poirier, D., Allmaras, S., McCarthy, D., Smith, M., Enomoto, F., "The CGNS System,,," AIAA Paper 98-3007, Reno, NV, January 1998.
 14. Poirier, D., "CGNS Standard for Aerodynamic Data,,," May 17, 2001, <http://www.cgns.org/>, (May 24, 2001).
 15. Bush, R.H., Power, G.D., Towne, C., "WIND: The Production Flow Solver of the NPARC Alliance,,," AIAA Paper 98-0935, Reno, NV, Jan. 1998.
 16. Towne, C., "WIND Documentation,,," April 18, 2001, <http://www.grc.nasa.gov/www/winddocs/>, (May 24, 2001).
 17. Spekrijse, S.P., Boerstael, J.W., Vitagliano, P.L., Kuyvenhoven, J.L., "Domain Modeling and Grid Generation for Multi-Block Structured Grids with Application to Aerodynamic and Hydrodynamic Configurations,,," Proceedings of the Software Systems for Surface Modeling and Grid Generation Workshop, R.E. Smith (Ed.), NASA Conference Publication 3143, p. 207, NASA Langley Research Center, Hampton, VA, 1992.
 18. Dannenhoffer, J.F. "Automatic Blocking for Complex Three-Dimensional Configurations,,," Surface Modeling, Grid Generation, and Related Issues in Computational Fluid Dynamics, NASA Conference Publication 3291, p. 123, Workshop Proceedings, NASA Lewis Research Center, May 1995.

Figures

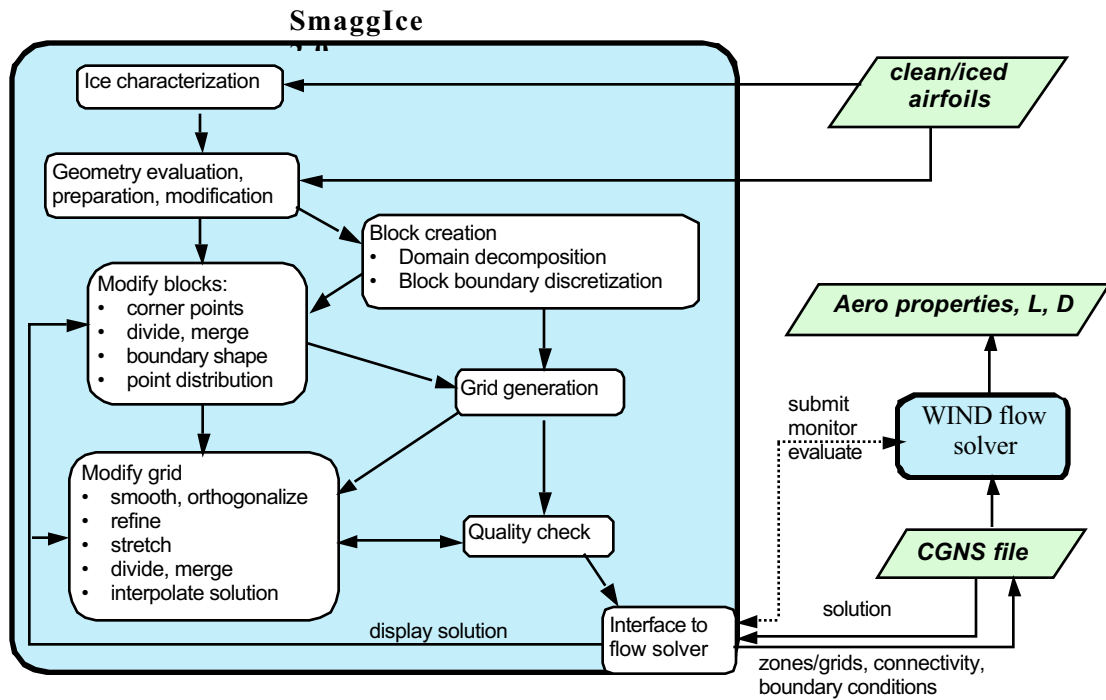


Figure 1. SmagIce 2D version 2.0 tool interaction and process flow

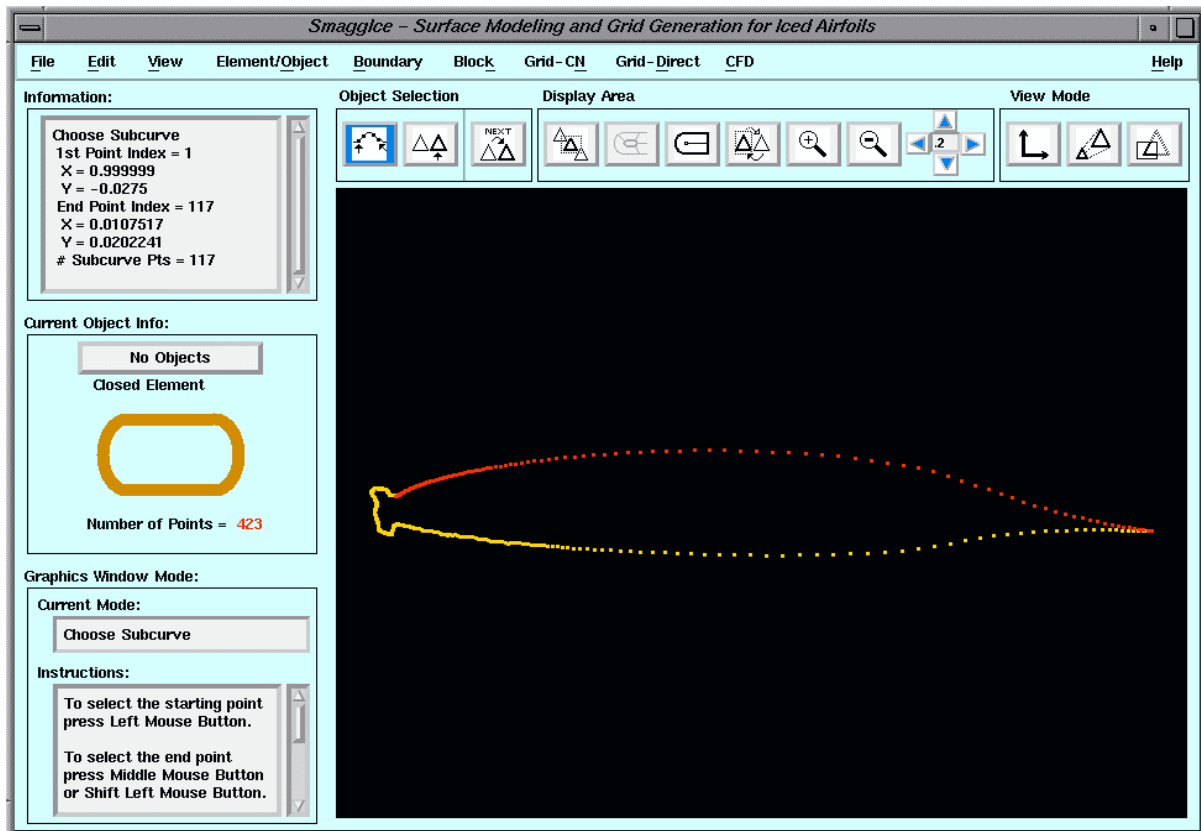


Figure 2. SmagIce 2D Main Window

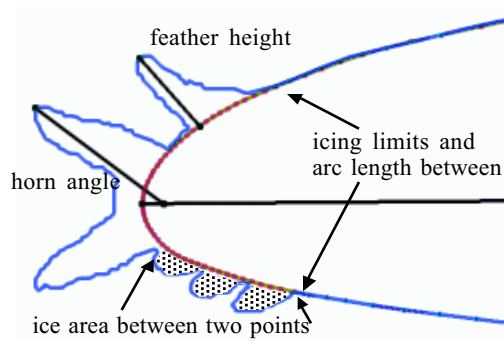


Figure 3. Ice shape characterization

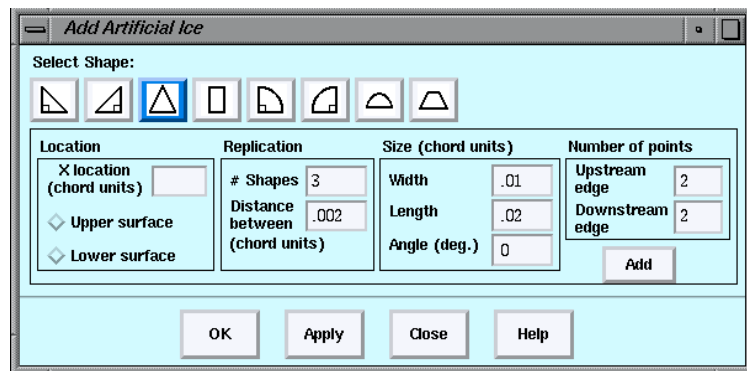


Figure 4. Artificial Ice Window

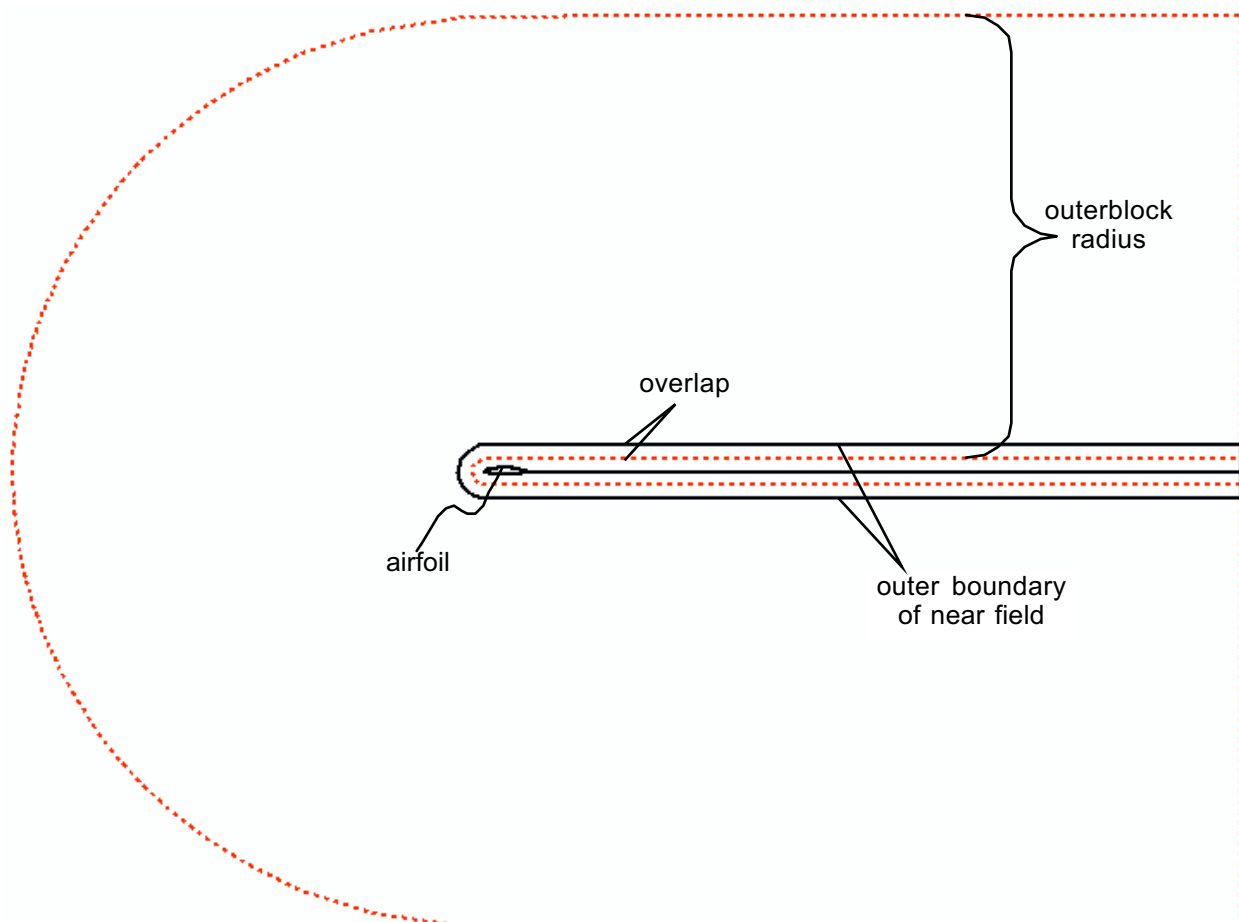


Figure 5. The outer block can overlap the near field, as shown here. The near field is further decomposed based on the ice shape and the chosen class. The viscous sublayer block is too narrow to be visible in this view.

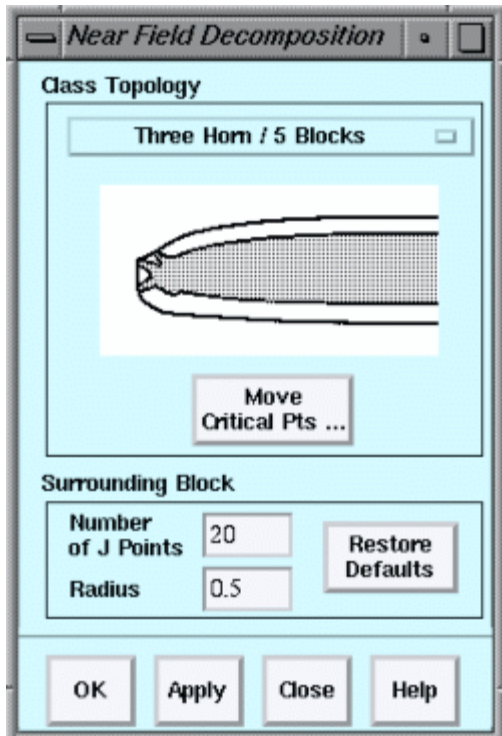


Figure 6. Near Field Decomposition Window

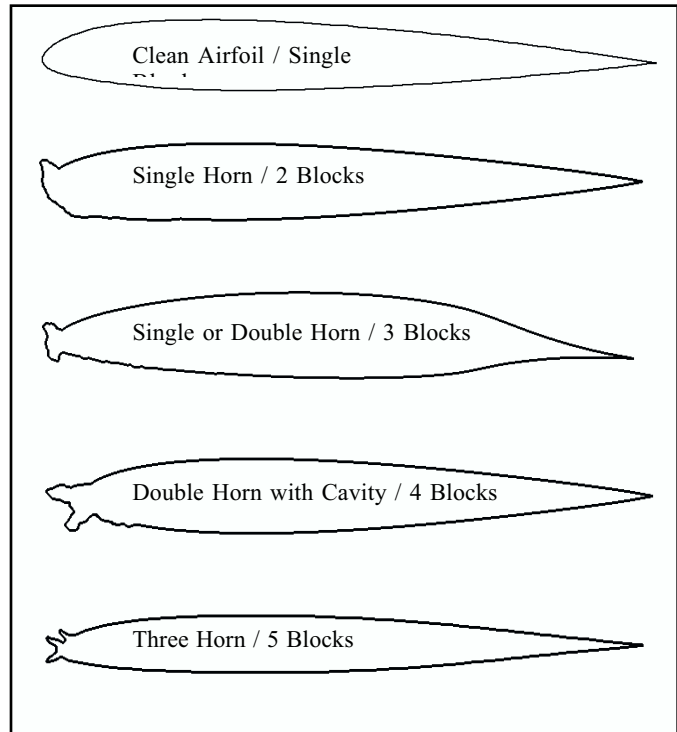


Figure 7. Ice shape classes

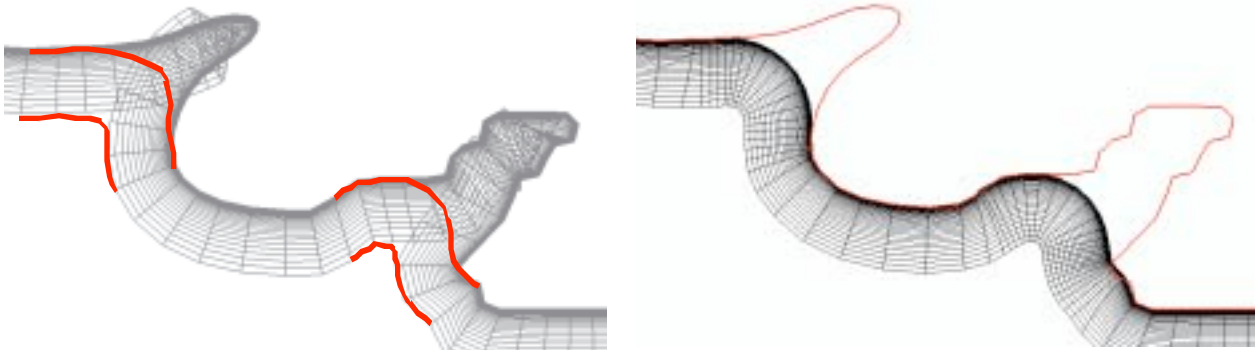


Figure 8. *left*: Viscous sublayer before removing tangles. *right*: Viscous sublayer after automatic untangling.

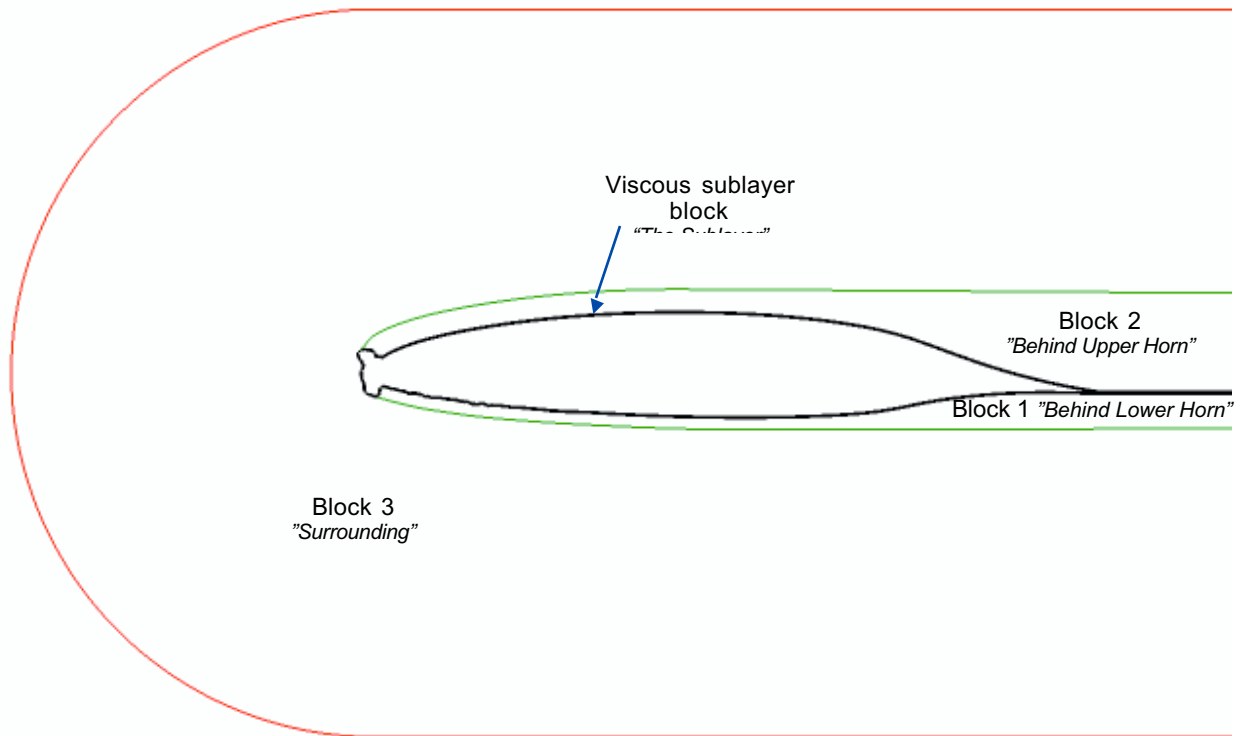


Figure 9. Close-up view of initial near-field blocking for an ice shape using class “Single or Double Horn / 3 Blocks,,”. The outer block is not shown. The viscous sublayer block is very narrow and here looks like a thick black line.

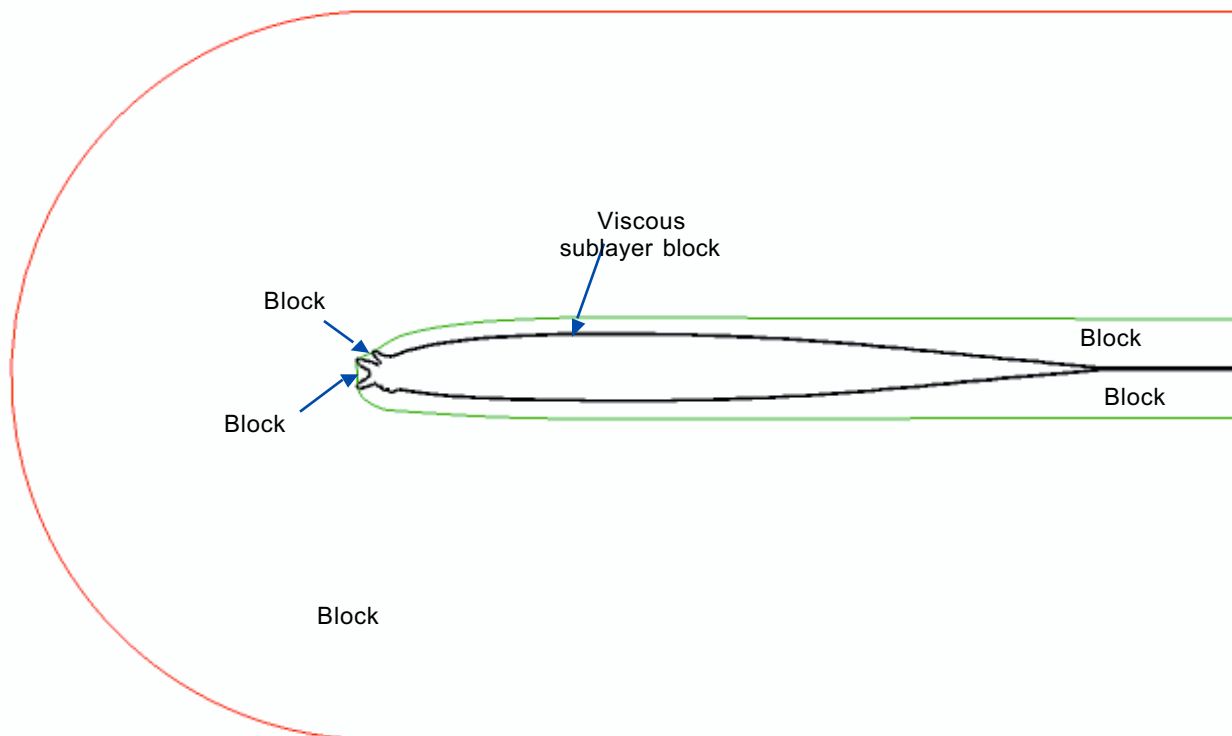


Figure 10. Close-up view of initial near-field blocking for an ice shape using class “Three Horn / 5 Blocks,,”.

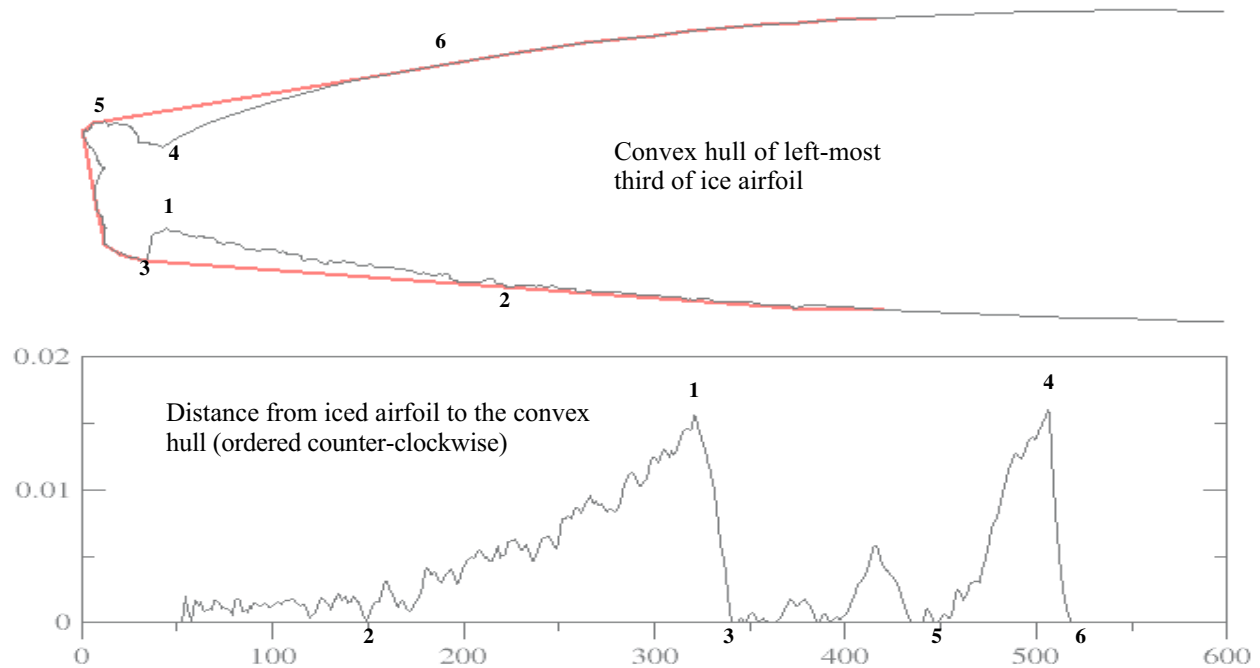


Figure 11. Critical point selection: In class “Single or Double Horn / 2 Blocks,, two blocks are needed in the near field. For the first block, the maximum distance at point (1) is found, as well as the points (2 and 3) on either side that have a distance of zero. Then that segment is removed from consideration, and the next maximum distance is found at point (4), etc. Notice that points (2 and 6) are not actually used in the domain decomposition, as the corners of those blocks will end up being located on the exit boundary far downstream.

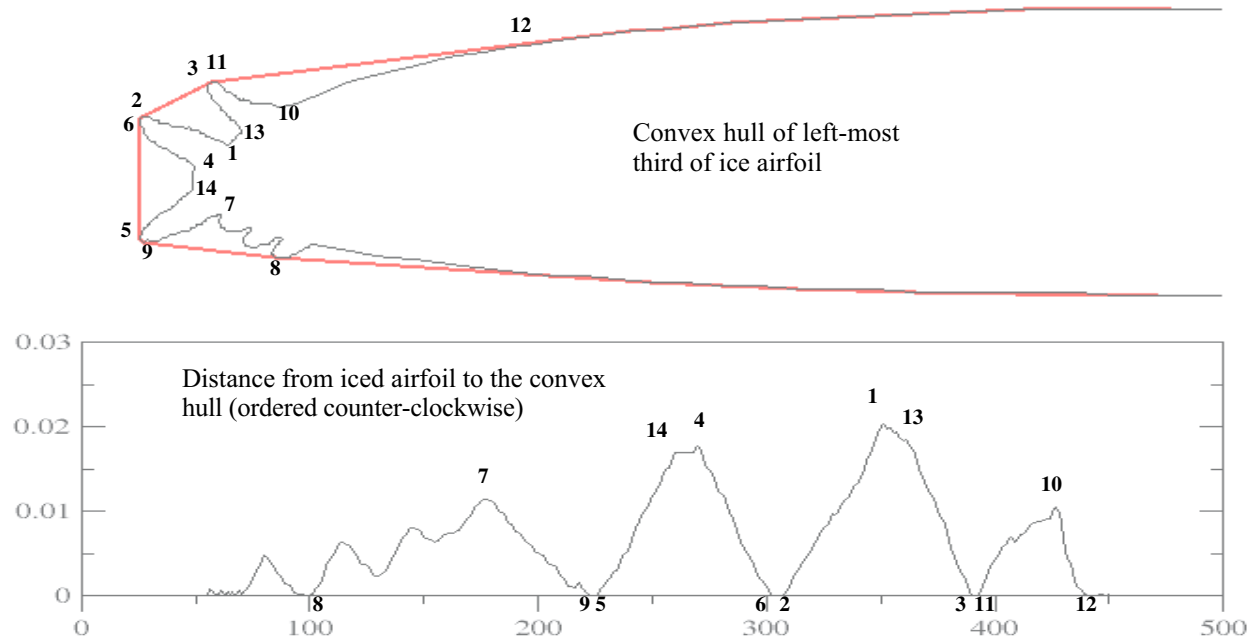


Figure 12. Critical point selection for class “Three Horn / 5 Blocks,,. In this case, four blocks are needed around the airfoil, so four segments (and their peaks: points 1, 4, 7, and 10) are found using the convex hull method. For each of the two blocks between horns (with critical points 1 and 4), an additional corner point must be selected. For those blocks, local maxima are found to set critical points 13 and 14. As in the case in Figure 10, the rightmost and leftmost points (8 and 12) are not used, since the corner points of those blocks will be on the exit boundary.